Detection of Mind Wandering using EEG Signals

A Project Report(Group_no--17)

Submitted for Minor Project (6CS191) of 6th Semester for partial fulfillment of the requirements for the award of the degree of

Bachelors of Technology
in
Computer science and Engineering

Submitted by

Prashant Thakur	**1706001**
Nehal Kumar	**1706049**
Atul Raj	**1706090**
Vaibhav Jaiswal	**1706137**

Under the Supervision of

Prof. Mukesh Kumar
Asst. Prof. CSE Department
NIT Patna

Department of Computer Science & Engineering

National Institute of Technology Patna
Patna-800005

Jan-July, 2020

राष्ट्रीय प्रौद्योगिकी संस्थान पटना
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

CERTIFICATE

This is to certify that **Prashant Thakur Roll No. 1706001, Nehal Kumar Roll No. 1706049, Atul Raj Roll No. 1706090, Vaibhav Jaiswal Roll No. 1706137** has carried out the Minor project (6CS191) entitled as **"Detection Of Mind Wandering using EEG Signals"** during his 6th semester under the supervision of **Prof. Mukesh Kumar,** Asst. Prof., CSE Department in partial fulfillment of the requirements for the award of Bachelor of Technology degree in the department of Computer Science & Engineering, National Institute of Technology Patna.

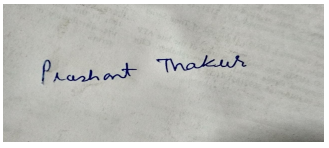
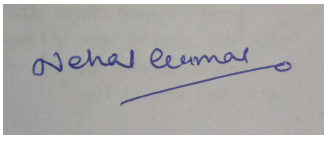
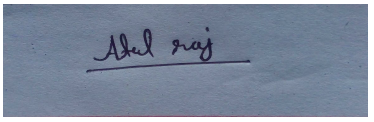
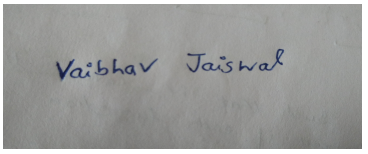
.....
Prof. Mukesh Kumar
Assistant Professor
CSE Department
NIT Patna

.....
Dr. J. P. Singh
Head of Department
CSE Department
NIT Patna

राष्ट्रीय प्रौद्योगिकी संस्थान पटना
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

DECLARATION

We students of 6th semester hereby declare that this project entitled "**Detection Of Mind Wandering using EEG Signals**" has been carried out by us in the Department of Computer Science and Engineering of National Institute of Technology Patna under the guidance of **Prof. Mukesh Kumar**, Department of Computer Science and Engineering, NIT Patna. No part of this project has been submitted for the award of degree or diploma to any other Institute.

Name	**Signature**
Prashant Thakur	
Nehal Kumar	
Atul Raj	
Vaibhav Jaiswal	

Place: **Date:**

NIT Patna **29/07/2020**

राष्ट्रीय प्रौद्योगिकी संस्थान पटना
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

ACKNOWLEDGEMENT

We would like to acknowledge and express my deepest gratitude to my mentor **Prof. Mukesh Kumar**, Assistant Professor, Computer Science & Engineering Department, National Institute of Technology Patna for the valuable guidance, sympathy and co-operation for providing necessary facilities and sources during the entire period of this project.

I wish to convey my sincere gratitude to the Head of Department and all the faculties of Computer Science & Engineering Department who have enlightened me during our studies. The faculties and cooperation received from the technical staff of Department of Computer Science & Engineering is thankfully acknowledged.

1. Prashant Thakur (Roll No. 1706001)
2. Nehal Kumar (Roll No. 1706049)
3. Atul Raj (Roll No. 1706090)
4. Vaibhav Jaiswal (Roll No. 1706137)

Contents

	PageNo.
I. Certificate	2
II. Declaration	3
III. Acknowledgement	4
IV. Contents	5
1. Abstract	6
2. About the Project	
1.1 Introduction	8
1.2 Mind Wandering Experiment	9
3. Implementation	10
4. Technologies and Learning's	
1.1 Python	14
1.2 MNE	14
1.3 EEG Signal Analysis and Preprocessing	15
1.4 Deep Learning	15
1.5 LSTM	16
1.6TensorFlow	16
1.7 Various Deep Learning Libraries	17
5. Result	18
6. Conclusion	19
7. References	20

1. ABSTRACT

Abstract—Mind wandering (MW) is a ubiquitous phenomenon which reflects a shift in attention from task-related to task-unrelated thoughts. There is a need for intelligent interfaces that can reorient attention when MW is detected due to its detrimental effects on performance and productivity. In this paper, we propose a deep learning model for MW detection using Electroencephalogram (EEG) signals. Specifically, we develop a channel-wise deep Long short-term memory neural network (LSTM) model to classify the features of focusing state and MW extracted from EEG signals. This is the first study that we know that employs LSTM to automatically detect MW using only EEG data. The experimental results on the collected dataset demonstrate promising performance with 92% accuracy.

2. ABOUT THE PROJECT

1.1 Introduction

Most people have had the experience of a shift in their focus from an attention-demanding task at hand, such as reading books or driving, to task-unrelated concerns. For instance, they may be thinking of events which happened in the past, may happen in the future, or never happen at all. This stimulus-independent experience, which is a series of imaginative thoughts intermittently during sustained attention tasks, is called mind wandering (MW) [1] . Although some research linked MW to increased creativity [2], other studies indicated that MW can lead to errors and a decrease in performance in various tasks[3] . Moreover, there are evidences that supports the correlation between MW and emotional disorders, such as neuroticism, alexithymia, and dissociation [4]. Given the ubiquity and negative consequences of MW, it is beneficial to detect when MW occurs and then intervene to restore attention to the task at hand. As an initial step in this direction, this paper proposes an automated model to detect momentary occurrences of MW.

MW detection is a relatively unexplored field. A majority of research have correlated oculometric measures such as blink rate [5], pupil diameter and response, and eye gaze [6] to MW. Recently, there is a trend to study MW using EEG signals. The cognitive results by previous known techniques are inconclusive, disputable and sometimes contradictory[7]. That is why finding a data-driven technique that can detect MW accurately, efficiently is vital.

Deep learning techniques have been vastly used to perform tasks such as making predictions and decisions. **Long short-term memory** (**LSTM**) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video).
We propose to employ LSTM for the automatic identification of MW only using raw EEG signals while using MNE for EEG preprocessing.

1.2 Mind Wandering Experiment

A. Subjects and procedure EEG data used in this research was acquired by Grandchamp et al. at the University of Toulouse (Data is available on: https://sccn.ucsd.edu/publicly available EEG data.html) [8]. Datasets were collected from two participants, a female age 25 and a male age 31 both without mental or neurological disorder. Here we summarized the data collection procedure. More details are available in the original paper . Participants sat in a dimly lit room in front of a computer screen. The task of the participants was to count backward each of their breath cycles (inhale/exhale) from 10 to 1 repeatedly. The mental state for the backward counting is referred as focusing state (FS). Participants had to press a mouse button whenever they realized they had lost track of their breath count (i.e. MW happens). They then filled a short questionnaire describing their mind wandering episode and pressed a button to resume the task. Each participant took ten 20-min sessions performing the above procedure. EEG data were captured using a 64 channel Biosemi Active Two system. The sampling rate was 1024 Hz. The data was recorded continuously through the whole sessions.

3. IMPLEMENTATION

The project was implemented on Google Colaboratory and Spyder using Python3 programming language. The project consist of two part

1. EEG signal preprocessing
2. Training model on preprocessed signal.

Preprocessing raw EEG signal

We have used MNE-Python package for signal preprocessing.

MNE-Python software is an open-source Python package for exploring, visualizing, and analyzing human neurophysiological data such as MEG, EEG, sEEG, ECoG, and more. It includes modules for data input/output, preprocessing, visualization, source estimation, time-frequency analysis, connectivity analysis, machine learning, and statistics.

Our raw signal consist of EEG signal obtained from a subject during ten different experiments on different days. Participant took ten 20-min sessions performing the above procedure.The raw data of .bdf format went to the following preprocessing.

- Interpolating raw data
- Bandpass Filter
- Downsampling
- Independent Component Analysis
- Epoching

WE have used ICA (I**ndependent component analysis**) for artifact removal
Independent component analysis (ICA) is a statistical and computational technique for revealing hidden factors that underlie sets of random variables, measurements, or signals.
Preprocessing the raw EEG data made sure that there were the least amount of errors during training. The raw data had been cleaned of various artifacts like heartbeat , eyeblink, eye movement, electrical noises and then downsampled and filtered so that only useful features would be readily

extracted by the deep learning model.

Separating preprocessed data into training and test data

We used the Tensorflow library for training our model.
TensorFlow is an open source software library for high performance numerical computation. We used
The 10 pre-processed EEG signals were saved in 10 separate .bdf file formats. We reload all the file and convert them in Pandas Dataframe. All the data frame were appended to form a single data frame consisting of 15651
Data point.

We divided our whole data into training and test data sets. We selected 85% of our datapoint as training data and last 15% as test data.

```
test_ind = len(df)- test_size
train = df.iloc[:test_ind]
test = df.iloc[test_ind:]
```

Normalization of data

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information. Normalization is also required for some algorithms to model the data correctly.
We used MinMaxScaler for normalization of our data. We trained our scaled by fitting it on training data and then using this scaler to transform both training and test data.

```
scaler.fit(train)
scaled_train = scaler.transform(train)
scaled_test = scaler.transform(test)
```

Min-max normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

Creating Time series Generator

Time series data must be transformed into a structure of samples with input and output components before it can be used to fit a supervised learning model.
This can be challenging if you have to perform this transformation manually. The Keras deep learning library provides the TimeseriesGenerator to automatically transform both univariate and multivariate time series data into samples, ready to train deep learning models.
For selecting the length of time series we analysed each signal plot and we came to the conclusion that the signal has a repeating pattern after every 250 data points.

```
generator = TimeseriesGenerator(scaled_train, scaled_train, length=length, batch_size=1).
```

Training model on Time series data
Given a sequence of numbers for a time series dataset, we can restructure the data to look like a supervised learning problem. We can do this by using previous time steps as input variables and use the next time step as the output variable.
Can be transformed into samples with input and output components that can be used as part of a training set to train a supervised learning model like a deep learning neural network.
This is called a sliding window transformation as it is just like sliding a window across prior observations that are used as inputs to the model in order to predict the next value in the series. In this case the window width is 250 time steps.

We used LSTM(Long Short Term Memory networks) deep learning model for training on our time series data.
Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior.

The network has a visible layer with 83 input, a hidden layer with 128 LSTM blocks or neurons, and an output layer with 128 value.

```
model = Sequential()
model.add(LSTM(128, activation='relu', input_shape=(length,
n_features)))
model.add(Dense(83))
model.compile(optimizer='adam', loss='mse')
```

The RELU activation function is used for the LSTM blocks. ReLU stands for rectified linear unit, and is a type of activation function. Mathematically, it is defined as y = max(0, x).

WE have used Adam as an optimizer and Mean squared error (MSE) as Loss function.

Optimizers are algorithms or methods used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses.
Adam is an adaptive learning rate method, which means, it computes individual learning rates for different parameters. Its name is derived from adaptive moment estimation, and the reason it's called that is because Adam uses estimations of first and second moments of gradient to adapt the learning rate for each weight of the neural network.

The function that is used to compute this error is known as Loss Function.
The mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value.
The network is trained for 20 epochs . We used early stop as callback to avoid overfitting of the model.
After training is complete we save our model as .h5 file format and scaler as .pkl file format.

```
model.save("drive/My Drive/saved
model/mindwandering_model5.h5")

import joblib

joblib.dump(scaler,'drive/My Drive/saved
model/mindwandering_scaler5.pkl')
```

Loading our trained model and scaler and evaluation

We can also phrase the problem so that multiple, recent time steps can be used to make the prediction for the next time step.
Our model take 250 timestamp as input and produce one next timestamp as output.
Each time stamp consist of 83 features. We run our trained model on whole test data set starting from last 250 timestamp of training data set till we generate whole test data set.
In this way we predict the who test data set as output of model.
We then apply inverse_transform on predicted output .

```
test_predictions =
scaler.inverse_transform(test_predictions)
```

Predicting Mind Wandering

We used our generated test_prediction after applying inverse_transform to predict the mind wandering state. The feature condition in our data set represents state of mind.
Feature condition has three value

1. Start count : It refers to timestamp when person start counting from beginning
2. Qsn start : It refers to timestamp when mind wandering occur and subject is given questionnaire to solve
3. Qsn End : It refers to time when questionnaire is completed

EEG signal was continuous at each period of timestamp.

We divide the output of the condition class into three class.

1. condition <1.5
2. condition >=1.5 and condition <=2.5
3. condition >2.5

When “condition” < 1.5 we label the output as ‘Start count’, when it lies between 1.5 and 2.5 we label it as ‘Qsn start’ and when it is greater than 2.5 we label it as ‘Qsn end’ .
Hence we are able to predict the wandering state of person based on his EEG signal and our trained model.

We compare our predicted and actual label to obtain the accuracy of our model.

4. Technologies and Learning’s

1.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.
Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

1.2 MNE

MNE-Python software is an open-source Python package for exploring, visualizing, and analyzing human neurophysiological data such as MEG, EEG, sEEG, ECoG, and more. It includes modules for data input/output, preprocessing, visualization, source estimation, time-frequency analysis, connectivity analysis, machine learning, and statistics.

1.3 EEG Signal Analysis and Preprocessing

The electroencephalogram (EEG) is a dynamic noninvasive and relatively inexpensive technique used to monitor the state of the brain.
The brain contains unique information in many regions at any given time. An EEG signal recorded with electrodes placed on the scalp consists of many waves with different characteristics. Arrays of electrodes are distributed over the entire scalp. The large amount of data recorded from even a single EEG electrode pair presents a difficult interpretation challenge. Signal processing methods are needed to automate signal analysis and interpret the signal phenomena.

1.4 Deep Learning

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.
Deep learning algorithms seek to exploit the unknown structure in the input distribution in order to discover good representations, often at multiple levels, with higher-level learned features defined in terms of lower-level features.
Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction.

1.5 LSTM

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior.

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.
LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.
The first step in our LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer."
The next step is to decide what new information we're going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values we'll update. Next, a tanh layer creates a vector of new candidate values, C~t, that could be added to the state. In the next step, we'll combine these two to create an update to the state.

1.6 TENSORFLOW

TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.
Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains.

1.7 Various Python Libraries

Various other Python library were used in this project like

1. **Numpy**
   NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays.
2. **Pandas**
   Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is

called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

3. **Matplotlib**
   Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
4. **Sklearn**
   Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.
5. **Joblib**
   Joblib is a set of tools to provide lightweight pipelining in Python. In particular:
   1. transparent disk-caching of functions and lazy re-evaluation (memoize pattern)
   2. easy simple parallel computing

5. Result

The proposed model was run on an online jupyter notebook environment provided by google called google colab.After the implementation of all the unit modules like preprocessing raw signal, normalizing preprocessed signal,training model and predicting output, integration process was carried out to integrate all the modules together. We selected different activation function, loss function no of layers in LSTM model. We performed different runs of the training module. The final model took several hours to train on the dataset.

The trained model was then tested with the splitted 15% of the dataset. Testing yielded an accuracy of around 92%. Thus this much high accuracy of the model can be interpreted as a successful model.

However both the training and testing dataset were from only one participant with many characteristics unique to that participant, hence this model needs to be developed and tested on more number of participants. For obtaining a good generalization performance, a diversity of data is required. Various performance metrics need to be included too for more analysis of the model.

There is a lot of scope for this project where it can be used in the medical field for detection of emotional disorders, such as neuroticism, alexithymia, and dissociation [4]. More technological advancement need to be made in area of easier recording and processing of EEG signal before successful use of this project in field of detecting MW that can lead to errors and a decrease in performance in various tasks in daily life[3].

6. Conclusion

During the course of 6 th semester while working on this project, We got to know a lot of domain specific knowledge related to EEG Signal collection and preprocessing .We learnt various method of signal preprocessing technique and artifact removal from signal. We also come to know about tools like MNE-Python , Tensorflow, ICA.

The main motivation is that if you can't beat artificial intelligence, maybe you can join it. This is where an experimental neural mesh technology currently being tested in animal models could be a major step forward. By injecting self-unfolding webs of electrodes into our heads and letting them unfold around the brain, we could interact with electronics around us using nothing more than a thought.

In our project we can continuously analyse EEG signal generated from brain and analyse it using our trained model and detect when Mnd wandering occur occurs and then intervene to restore attention to the task at hand.

The experimental results showed that our model can achieve the classification accuracy up to 92%.The initial success of our model warrants our future study to apply the same deep learning approach on new datasets for automated MW detection.

REFERENCES

[1] C. Braboszcz and A. Delorme, “Lost in thoughts: Neural markers of low alertness during mind wandering,” Neuroimage, vol. 54, no. 4, pp. 3040–3047, 2011.

[2] G. A. Shaw and L. M. Giambra, “Task-unrelated thoughts of college students diagnosed as hyperactive in childhood,” Dev. Neuropsychol., vol. 9, no. 1, pp. 17–30, 1993

[3] J. C. Mcvay, M. J. Kane, and T. R. Kwapil, “Tracking the train of thought from the laboratory into everyday life: An experience sampling study of mind wandering across controlled and ecological contexts,” Psychon Bull Rev, vol. 16, no. 5, pp. 857– 863, 2009.

[4] S. G. Jensen, C. G., Niclasen, J., Vangkilde, S. A., Petersen, A., & Hasselbalch, “General inattentiveness is a long-term reliable trait independently predictive of psychological health: Danish validation studies of the Mindful Attention Awareness Scale,” vol. 28, no. 5, pp. 70–87, 2016.

[5] D. Smilek, J. S. A. Carriere, and J. A. Cheyne, “Out of mind, out of sight: eye blinking as indicator and embodiment of mind wandering.,” Psychol. Sci. a J. Am. Psychol. Soc. / APS, vol. 21, no. 6, pp. 786–789, 2010.

[6] P. W. Kandoth and D. H. Deshpande, “Automatic gaze-based user independent detection of mind wandering during computerized reading,” J. Postgrad. Med., vol. 16, no. 4, pp. 205–208, 1970.

[7] R. Grandchamp et al., “Pupil dilation during mind wandering,” vol. 9, no. 3, p. 18298, 2011.

[8] R. Grandchamp, C. Braboszcz, and A. Delorme, “Oculometric variations during mind wandering,” Front. Psychol., vol. 5, pp. 1– 10, 2014.

[9] https://mne.tools/stable/index.html

[10] https://colab.research.google.com/

[11] https://www.kaggle.com/ashishshaji/eeg-emotion-lstm